MDMapper Results for OAEI 2024

Xianhao Liu[1,2], Michael R. Hansen[1] and Jesper Grode[2]

[1]*Technical University of Denmark, 2800 Kgs-Lyngby, Denmark*
[2]*Stibo Systems A/S, Axel Kiers Vej 11, 8270 Højbjerg, Denmark*

Abstract
This paper presents the results of the participation of MDMapper [1] in OAEI 2024. The tool MDMapper is a matching system under development, designed mainly for master data models, but it also supports simple ontology matching tasks. Master data models are typically simple hierarchical classification system, where basic concepts have rich descriptions involving types and units, for example.

The tool combines mature ontology matching techniques with newly proposed methods. In particular, MDMapper combines string-based and transformer-based similarity measures to compute an overall similarity matrix. It initially identifies high confidence correspondences $(c_1, c_2, =)$ (reads: "c_1 and c_2 are equivalent") and then progressively discovers new correspondences while reducing the matching space using a novel conflict-based restriction management approach. As a first-time participant of OAEI, MDMapper participated in the Anatomy, Conference, and Multifarm tracks.

Keywords
Ontology Matching, Master Data Models, Similarity Measurement

1. Presentation of the system

The long-term goal of this work is to achieve highly-automated matching of models and conversion of data originating from different agents (data suppliers) in master data management systems [2]. An organization's master data represents a major asset for the organization and it is integral to maintain a trustworthy, high-quality repository of data.

1.1. Purpose, general statement

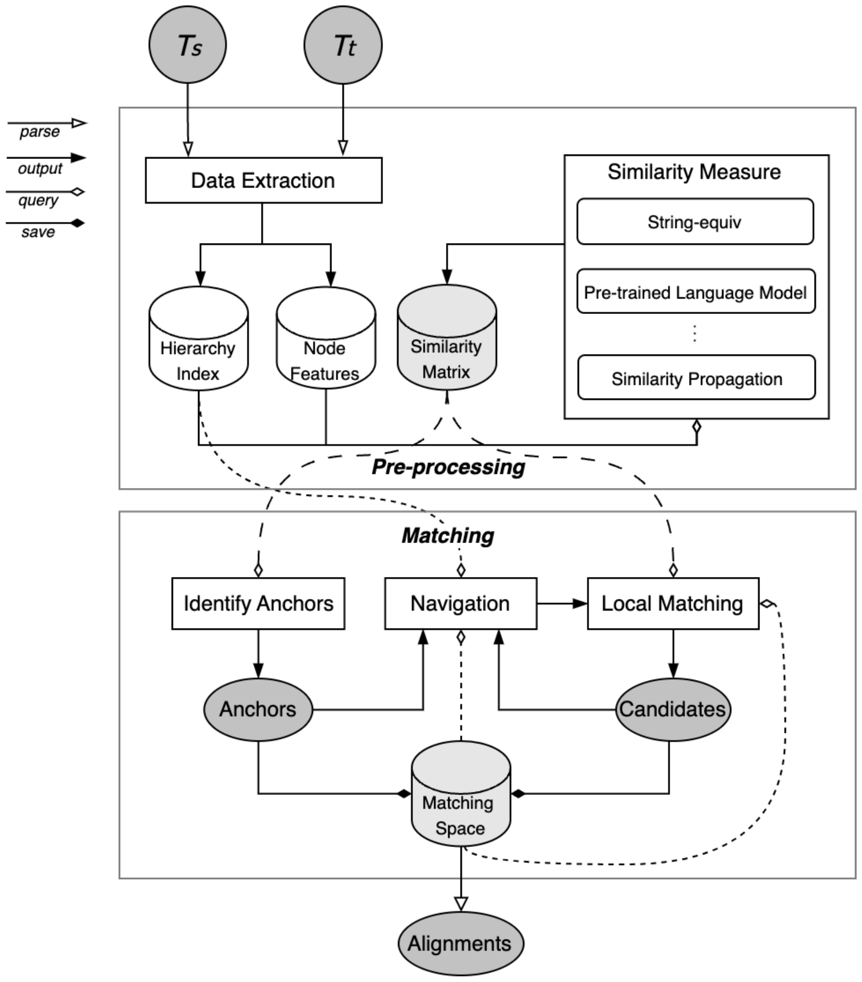
The tool MDMapper is specifically designed to handle matching of master data models. Such models (or MDM ontologies) can be viewed as simply ontologies with hierarchical structures and detailed attribute descriptions. Since it is important to maintain high-quality master data, the computed alignment for two ontologies must form a consistent set of correspondences. To achieve this, we are especially inspired by the concepts and techniques of LogMap [3] focusing on consistency. Furthermore, we use a conflict-based restriction management approach when exploring new correspondences that is based on Hansen et al. [4].

The current and first version of MDMapper can handle hierarchical ontologies and supports the matching of classes. The matching of attributes in master data is supported to a limited

OM-2024: The 19th International Workshop on Ontology Matching collocated with the 23rd International Semantic Web Conference (ISWC 2024), November 11th, Baltimore, USA.

xianliu@dtu.dk (X. Liu); mire@dtu.dk (M. R. Hansen); jnrg@stibosystems.com (J. Grode)
0009-0001-7410-0616 (X. Liu); 0000-0002-4775-4622 (M. R. Hansen); 0000-0002-1715-4597 (J. Grode)

Figure 1: The architecture of MDMapper.



extent. The ultimate goal is to expand the capabilities to include matching of classes, properties, and instances for ontologies in general.

1.2. Specific techniques used

The tool takes a source and a target ontology denoted by T_s and T_t, respectively, as input. These ontologies are processed in two phases called *pre-processing* and *matching*. The main structure of MDMapper [1] is shown in Figure 1.

The outcome of the pre-processing are similarity matrices for classes and properties. These matrices are input to the matching phase that produces an alignment in the form of a set of correspondences, including relationships involving $\equiv, \leq$ and $\geq$.

The main components of the pre-processing phase are:

Data extraction: We use *owlready2* [5] for parsing the metadata of ontologies. Specifically, labels, structures, annotations, and synonyms are extracted and used in the matching process. To unify the format of strings and text, various string normalization techniques are applied to remove stop words, handle special characters and other text anomalies.

Similarity measures: MDMapper computes a similarity matrix for both classes and properties. To capture similarity from different perspectives, multiple similarity measures are combined. We use ISUB [6] to extract lexical similarity and Sentence-BERT [7] with the all-MiniLM-L12-v2 model[1] to compute semantic similarity. By linearly combining the scores from these matchers, we obtain a composited similarity matrix, with values ranging from 0 to 1.

Similarity propagation: A so-called *Heuristic overall similarity measure* is applied to estimate the similarity between sets of entities, that is, sets containing classes or properties. This technique is used to combine the features of properties into the similarity of classes and to propagate similarities from subclasses to parent classes in a bottom-up manner.

The main components of the matching phase are:

Identifying anchors: *High-confidence initial equivalences* serve as anchors for the subsequent matching process. For an anchor $(c_s, c_t, =)$, where $c_s \in T_\mathrm{s}$ and $c_t \in T_\mathrm{t}$, the following conditions must be met:

1. The similarity score between c_s and c_t must exceed a given threshold.
2. The classes c_s and c_t must mutually be most similar to each other, that is,
   - there is no c_t' so that c_s and c_t' have a higher similarity than that of c_s and c_t, and
   - there is no c_s' so that c_s' and c_t have a higher similarity than that of c_s and c_t.
3. There should be a margin between the similarity of c_s and c_t and other similarities for c_s and c_t.

Relation derivation: For a given pair of classes c_s and c_t, new relations between their respective subclasses are identified in a bottom-up manner. These subclass-level relations are then used to derive a relation between c_s and c_t. The possible relations are equivalence, specialization, generalization, disjoint and partial overlap.

Relation-based navigation: The derived relations are used in the search for new candidate equivalences. If, for example, we consider (c_s, c_t) and know that $(c_s, c_t, \leq)$, then the navigation step "moves to" $(\mathrm{p}arent(c_s), c_t)$.

Matching space: During the matching phase, a *matching space* is maintained that keeps track the current alignment and the candidate correspondences that are consistent with the current alignment. The matching space works together with the navigation functions to enable local matching.

[1]https://huggingface.co/sentence-transformers/all-MiniLM-L12-v2

Local Matching: For a given candidate pair (c_s', c_t') coming from the relation-based navigation, the local matching process exploits the matching space to narrow the matching scope as much as possible.

1.3. Adaptations made for the evaluation

The MELT system [8] was used to package MDMapper to SEALS.

For all OAEI tracks, we applied fixed parameters optimized for the Anatomy track.

In the identify-anchor step:

- the threshold was set at 0.865,
- the margin gap to alternative candidates was set to 0.01, and
- the step size was set to 0.05,

where the step size controls the scale of threshold decay during the subsequent matching process. Specifically, if the parent entities of a mapping candidate are found to be equivalent, a lower threshold is applied to accept the equivalence of the candidate entities.

Conference ontologies include rich properties that were not parsed or utilized in this iteration of MDMapper.

2. Results

Since MDMapper is designed specifically for MDM ontology matching, it only participated in OAEI tracks involving simple ontologies, see Table 1. The results for the Anatomy and Conference tracks were promising; however, the performance on the Multifarm track was not meaningful due to the current lack of support for multilingual inputs.

Table 1
Results of MDMapper in OAEI 2024

Track	Runtime (s)	Precision	Recall	F1-Measure	Recall+
Anatomy	121	0.926	0.881	0.903	0.703
Conference	-	0.66	0.53	0.59	-
Multifarm	-	0.199	0.017	0.032	-

2.1. Anatomy

MDMapper delivered a solid performance in the 2024 OAEI anatomy track, ranking second in F-measure (0.903), just behind Matcha (0.941). It generated 1,441 correspondences, comparable to top systems like LogMapBio (1,549) and Matcha (1,485). Its precision (0.926) is decent, but not as highly competitive compared to other systems.

MDMapper's recall (0.881) and recall+ (0.703) were strong, indicating its ability to capture a broad range of correspondences.

The MDMapper runtime of 121 seconds on the Anatomy dataset is longer than that of the fastest systems, but remains within a reasonable range. However, the process did not yield a coherent alignment as expected, with some inconsistencies still present within the correspondences. This incoherence may be due to shortcomings in the anchor identification step, which lacked verification through a consistency check.

Overall, MDMapper performed well in alignment quality, especially in recall and F-measure, making it a competitive system in the anatomy track. Future improvements could focus on ensuring coherent alignments and enhancing precision without compromising overall performance.

2.2. Conference

MDMapper achieved a competitive result in the 2024 OAEI conference track, ranking third in F-measure (0.59). Its recall (0.53) was the third highest among all systems, although its precision (0.66) was lower compared to several others. Conference ontologies include rich properties that were not parsed or utilized in this iteration of MDMapper. Future improvements should focus on incorporating these properties into the matching process, leveraging them as features and correspondences at the property level.

From the **Results of Evaluation for the Conference track within OAEI 2024**:

> “MAMapper exhibits stable recall across all metrics, from 0.55 in the sharp to 0.64 in both the discrete and continuous evaluations. However, its precision drops slightly from 0.71 in sharp to 0.66 in the discrete setting and 0.69 in the continuous setting. This suggests that while MAMapper is effective at recalling uncertain matches, it struggles to assign high confidence to them, which negatively impacts its precision.”

Overall, MDMapper performed well in the conference track, with room for improvement by incorporating property-level matching.

2.3. Multifarm

MDMapper performed poorly on the Multifarm track, with results largely insignificant due to its current lack of support for multilingual inputs. This limitation severely impacted its effectiveness in this track.

3. Conclusions

MDMapper is specifically designed to address master data management (MDM) matching problems, where the structure is simple but rich in attributes. Therefore, its optimal use is in ontology matching tasks with similar straightforward structures. While participating in OAEI 2024, it was most suited for tracks with such characteristics, such as the Anatomy and Conference tracks.

As a new system and a first-time participant in OAEI, MDMapper currently lacks several implementations and adaptations tailored for the various OAEI tracks. Features such as support for properties and instances, which have not yet been incorporated, could significantly enhance its performance in future evaluations.

Despite these limitations, MDMapper achieved a commendable result, particularly in the Anatomy and Conference tracks, showing strong potential as it continues to evolve.

Acknowledgement

This study was funded by Innovation Fund Denmark (grant number 2050-00004B).

References

[1] X. Liu, J. Grode, M. R. Hansen, Mdmapper: A framework for aligning master data models using ontology matching techniques (2024).

[2] T. Kumar Das, M. R. Mishra, A study on challenges and opportunities in master data management, International Journal of Database Management Systems 3 (2011) 129–139.

[3] E. Jiménez-Ruiz, B. Cuenca Grau, Logmap: Logic-based and scalable ontology matching, in: The Semantic Web–ISWC 2011: 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I 10, Springer, 2011, pp. 273–288.

[4] M. R. Hansen, X. Liu, J. Grode, Consistent alignments for simple ontologies in the digital information supply chain, in: The Practice of Formal Methods, Springer, 2024, pp. 175–194. Chapter 9.

[5] J.-B. Lamy, Owlready: Ontology-oriented programming in python with automatic classification and high level constructs for biomedical ontologies, Artificial intelligence in medicine 80 (2017) 11–28.

[6] G. Stoilos, G. Stamou, S. Kollias, A string metric for ontology alignment, in: The Semantic Web–ISWC 2005: 4th International Semantic Web Conference, ISWC 2005, Galway, Ireland, November 6-10, 2005. Proceedings 4, Springer, 2005, pp. 624–637.

[7] N. Reimers, I. Gurevych, Sentence-bert: Sentence embeddings using siamese bert-networks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2019. URL: https://arxiv.org/abs/1908.10084.

[8] S. Hertling, J. Portisch, H. Paulheim, MELT - matching evaluation toolkit, in: International conference on semantic systems (SEMANTICS), 2019, pp. 231–245.